A Fast Block Sum Pyramid Algorithm

빠른 블록 합 피라미드 알고리즘

정수목
삼육대학교 컴퓨터과학과

Soo-Mok Jung (jungsm@syu.ac.kr)
Dept. of Computer Science, Sahmyook University

중심어 : 움직임 추정, 블록 정합, 블록 합, 피라미드 알고리즘

Keyword : Motion Estimation, Block Matching, Block Sum Pyramid Algorithm

요 약

본 논문에서는 비디오코딩의 움직임 추정을 위한 빠른 블록 합 피라미드 알고리즘(FBSPA: Fast Block Sum Pyramid Algorithm)을 제안하였다. 제안된 알고리즘은 블록 합 피라미드 알고리즘, 블록 정합 움직임 추정을 위한 효율적인 다단계 연속 제거 알고리즘, 움직임 벡터 추정을 위한 빠른 알고리즘에 기초하고 있다. 제안된 알고리즘은 블록 합 피라미드 알고리즘의 연산량을 감소시키기 위하여 블록 합 피라미드 알고리즘에 부분 왜곡 제거 기법을 적용하였다. 제안된 방안이 블록 합 피라미드 알고리즘의 연산량을 최대 2.9% 감소시킬 수 있음을 실험을 통하여 확인하였다.

Abstract

In this paper, a Fast Block Sum Pyramid Algorithm (FBSPA) is presented for motion estimation in video coding. FBSPA is based on Block Sum Pyramid Algorithm(BSPA), Efficient Multilevel Successive Elimination Algorithms for Block Matching Motion Estimation, and Fast Algorithms for the Estimation of Motion Vectors. FBSPA reduces the computations for motion estimation of BSPA 2.9% maximally using partial distortion elimination(PDE) scheme.

I. Introduction

There is considerable temporal redundancy in consecutive video frames. Motion estimation and compensation techniques have been widely used in image sequence coding schemes to remove temporal redundancy. The accuracy and efficiency of motion estimation affects the efficiency of temporal redundancy removal.

Motion estimation methods are classified into two classes block matching algorithms (BMA)[1],[2] and pel-recursive algorithms (PRA)[3]. Owing to their implementation simplicity, block matching algorithms have been widely adopted by various video coding standards such as CCITT H.261[4], ITU-T H.263[5], and MPEG[6]. In BMA, the current image frame is partitioned into fixed-size rectangular blocks. The motion vector for each block is estimated by finding the best matching block of pixels within the search window in the previous frame according to matching criteria.

Although Full Search algorithm(FSA) finds the optimal motion vector by searching exhaustively for the best matching block within the search window, its high computation cost limits its practical applications. To reduce computation cost of FSA, many fast block matching algorithms such as three step search[7], 2-D log search[2], orthogonal search[8], cross search[9], one-dimensional full search[10], variation of three-step search[11],[12], unrestricted center-biased diamond search[13], a fast block matching algorithm using mean absolute error of neighbor search point and search region reduction[14] etc. have been developed. As described in [15], these algorithms rely on the assumption that the motion- compensated residual error surface is a convex function of the displacement motion vectors, but this assumption is rarely true [16]. Therefore, the best match obtained by these fast algorithms is basically a local

접수번호 : #030729-001
접수일자 : 2003년 7월 29일, 심사완료일 : 2003년 10월 21일

*교신저자 : 정수목, email : jungsm@syu.ac.kr

optimum.

Without this convexity assumption, Successive Elimination Algorithm(SEA) proposed by Li and Salari[17] reduces the computation cost of the FSA. To reduce the computation cost of SEA, Block Sum Pyramid Algorithm (BSPA)[18] and Multilevel Successive Elimination Algorithm (MSEA)[19] were proposed. Our research team proposed Efficient Multilevel Successive Elimination Algorithms for Block Matching Motion Estimation (EMSEA)[20] to reduce the computation cost of MSEA. The partial distortion elimination scheme of EMSEA was improved and then the improved scheme was applied to BSPA in FBSPA. The motion estimation accuracy of FBSPA is identical to that of FSA and the computation cost of BSPA is reduced using FBSPA.

II. Block Sum Pyramid Algorithm

The SEA achieves the same estimation accuracy as the FSA while requiring less computation time. In SEA, the displacement vector of the corresponding block in the previous frame is used as the initial motion vector for the present template block[21]. The SEA uses the sum norm of a block as a feature to eliminate unnecessary search points. The sum norm of a block B of size NxN is defined as

$$S_B = \sum_{i=1}^{N}\sum_{j=1}^{N} |B(i,j)| \quad (1)$$

where B(i, j) is the gray level of the (i, j)th pixel of block B. Let ST be the sum norm of the template block T, Sx be the sum norm of a candidate matching block X, and curr_MADmin be the current minimal MAD during the search process. Let MAD(T, X) be the MAD between T and X and is defined as

$$\mathrm{MAD}(T,X) = \sum_{i=1}^{N}\sum_{j=1}^{N} |T(i,j) - X(i,j)| \quad (2)$$

where T(i, j) and X(i, j) represent the gray values of the (i, j)th pixels of T and X. The equation (3) was verified in [17]:

$$\mathrm{MAD}(T,X) \geq \sum_{i=1}^{N}\sum_{j=1}^{N} |S_T - S_X| \quad (3)$$

Based on the above inequality, the SEA discards each candidate matching block X with |ST - SX| ≥ curr_MADmin, which can save a lot of search time. Block Sum Pyramid Algorithm can eliminate those impossible matching blocks by exploiting the sum pyramid structure of a block. An image pyramid is a hierarchical data structure originally developed for image coding [22]. Assume that each block is of size NxN with N=2n. Then, for each block X, a pyramid of X can be defined as a sequence of blocks { X0, ... , Xm-1, Xm, Xm+1, ... , Xn} with Xm-1 having size 2m-l x 2m-l and being a reduced-resolution version of Xm as shown in Fig. 1.

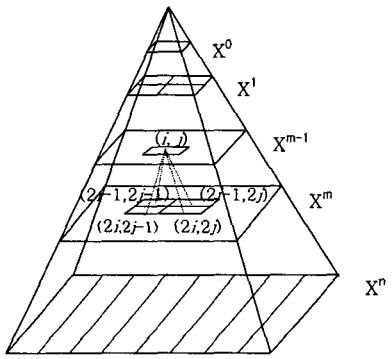

Fig. 1. A pyramid data structure

Note that X0 has only one pixel. A pyramid data structure can be formed by successively operating over 2x2 neighboring pixels on the higher levels. That is, the value of a pixel Xm-1(i, j) on level m-1 can be obtained from the values of the corresponding 2x2 neighboring pixels Xm(2i-1, 2j-1), Xm(2i-1, 2j), Xm(2i, 2j-1), and Xm(2i, 2j) on level m as shown in equation (4).

$$X^{m-1}(i,j) = X^m(2i-1,2j-1) + X^m(2i-1,2j) + X^m(2i,2j-1) + X^m(2i,2j)\} \quad (4)$$

For two blocks X and Y, let MADm(X, Y) be MAD(Xm, Ym), i.e.,

$$\mathrm{MAD}^m(X,Y) = \sum_{j=1}^{2^m}\sum_{h=1}^{2^m} |X^m(j,h) - Y^m(j,h)| \quad (5)$$

where Xm(j, h) and Ym(j, h) represent the values of the (j, h)th pixels on Xm and Ym respectively. Thus, on the top level, MAD0(X, Y)=|SX - SY|. From the above definition, the

following theorem holds.

$$MAD(X,Y) \geq MAD^{n-1}(X,Y) \geq MAD^{n-2}(X,Y) \geq \ldots \geq MAD^{0}(X,Y) \quad (6)$$

Block Sum Pyramid Algorithm uses the above theorem (6). The Block Sum Pyramid Algorithm first constructs the sum pyramid of every block that corresponds to a search position in the previous frame. To search for the best matching block of a template block T, the sum pyramid of T is established. Then, the MAD between T and the block with displacement vector (0, 0) is evaluated, and this value is considered as the current minimum MAD(symbolized as curr_MADmin). For any other search block X, the algorithm first checks the MAD on the top level, MAD0(T, X). If the calculated MAD0(T, X) (symbolized as cal_MAD0(T, X)) meets the equation (7) then this block can not become the best matching block. So, this block can be eliminated. If MAD0(T, X) does not meet the equation (7), the MAD on the first level is checked.

$$curr\_MAD_{min} \leq cal\_MAD^{0}(X,Y) \quad (7)$$

If the calculated MAD1(T, X) meets the equation (8), for the same reason above, this block can be eliminated. If it is not, the second level is tested.

$$curr\_MAD_{min} \leq cal\_MAD^{1}(X,Y) \quad (8)$$

The process is repeated until this block is eliminated or the bottom level is reached. If the bottom level is reached, then MAD(T, X) is calculated and checked. If MAD(T, X)< curr_MADmin, the current minimum distortion curr_MADmin is replaced with MAD(T, X). Block Sum Pyramid Algorithm can eliminate many search blocks without evaluating their MADs. Assume that the size of the image frame is WxH.

For each level of the pyramid, calculation of the sum of 2x2 neighboring pixels requires 3(W-1)(H-1) additions. However, using the idea for fast calculation of the sum norm developed in [17], the complexity can be reduced to be (2W-1)(H-1) additions for each level. If the block size is 16x16, i.e., N=16, the overhead for constructing the sum pyramid is 4(2W-1)(H-1). Since there are (W/N)(H/N) template blocks in an image frame, the computation overhead for each template block is expressed as equation (9).

$$4(2W-1)(H-1)/[(W/N)(H/N)] = N^2(8-4/W-8/H+4/WH) \approx 8N^2 \quad (9)$$

III. Fast Block Sum Pyramid Algorithm

The BSPA speeds up the process of finding the best motion vector by eliminating impossible candidate vectors before their MADs are computed. Partial distortion elimination (PDE)[21] can be used in with BSPA to reduce the computation further for these vectors where the norm |T(i,j)-X(i,j)| must actually be computed. PDE is an effective speed up technique used in vector quantization to find a best reconstruction vector from a set of vector code words.

We observe that since all of the terms in the equation (2) are positive, if at any point the partially evaluated sum exceeds the current minimum MAD(curr_MADmin), that candidate block X cannot be the best matching block and the remainder of the sum does not need to be calculated. While it is not efficient to test the partial sum against the curr_MADmin every additional term is added, a reasonable compromise is to perform the test after N times additions when block size is NxN as shown in Fig. 2. So, the maximum number of PDE test is N.

In BSPA, MAD is calculated all the NxN pixels and then the calculated MAD is compared with curr_MADmin.

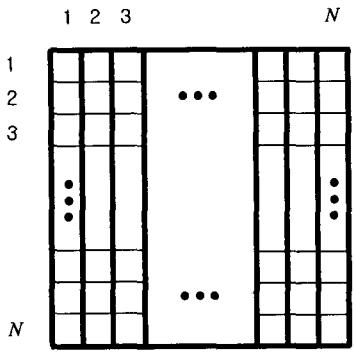

Fig. 2. PDE test in MAD calculation

FBSPA procedure is as follows:

step 1 select an initial search block within the search window in the previous frame.

step 2 calculate the motion vector (MV) and the MAD at the selected search block. these MV and MAD become he current temporary motion vector and the current minimum MAD respectively (temp_MV=MV, curr_MADmin= MAD)

step 3 select another search block among the rest of the search blocks

step 4.0 calculate the MAD0(T, X) at the selected search block. if (curr_MADmin $\leq$ MAD0(T, X) goto step6

step 4.1 calculate the MAD1(T, X) at the selected search block. if (curr_MADmin $\leq$ MAD1(T, X)) goto step6

.

.

.

step 4.(n-1) calculate the MAD(n-1)(T, X) at the selected search block.

if (curr_MADmin $\leq$ MAD(n-1)(T, X)) goto step6

step 4.n calculate the MAD at the selected search block for N pixels.

if (curr_MADmin $\leq$ MAD) goto step6

else if MAD is not calculated for all pixels(NxN) then goto step 4.n

step 5 curr_MADmin=MAD

calculate the motion vector at the selected search block. This motion vector becomes the current temporary motion vector (temp_MV=MV)

step 6 if (all the search blocks in the search window are not tested?) goto step3

step 7 the optimum motion vector =temp_MV,

the global minimum MAD= curr_MADmin

IV. Experimental Results of Fast Block Sum Pyramid Algorithm

The experiments were perormed on several typical QCIF(176x144) test sequences in the framework of the H.263 such as "grandma.qcif", "suzie.qcif", "claire.qcif". We tested 100 frames of the sequences. The block (Y component of the macroblock in H.263) size and the size of motion vector search windows for full pixel searching are 16x16 pixels (N=16) and 31x31 pixels (M=15) respectively and only integer values for the motion vectors were considered.

Experimental results are shown in table 1. In table 1, m.e. means matching evaluation that require MAD calculation, avg. # of rows means the number of calculated rows in the MAD calculation before partial distortion elimination. Overhead(in rows) is the sum of all the computations except MAD calculation. "in rows" means that the computations are represented in order of 1 row MAD computations.

It is important to notice that with the BSPA, the efficiency of the procedure depends on the order in which the candidate motion vectors are searched, and that the most promising candidates should be tested first. This eliminates the maximum number of candidates. In our experiment, we used spiral search.

The FBSPA which incorporates the BSPA with PDE, reduces the computations of BPSA by 2.9%, 2.8%, 2.2% for grandma.qcif, suzie.qcif, and claire.qcif respectively.

Table 1. The computations of FBSPA

Algo-rithm	Test sequence	Avg. # of m.e./frame	Avg. # of rows/m.e	Overhead (in rows)	Total (in rows)	Compu-tations reduction
BSPA	grandma	350.7	16.00	29,749.3	35,360.5	
	suzie	607.1	16.00	27,620.5	37,334.1	
	claire	228.3	16.00	17,920.8	21,573.6	
FBSPA	grandma	350.7	13.12	29,749.3	34,350.5	2.9%
	suzie	607.1	14.25	27,620.5	36,271.7	2.8%
	claire	228.3	13.89	17,920.8	21,091.9	2.2%

V. Conclusions

A Fast Block Sum Pyramid Algorithm based on BSPA has been proposed to reduce the computations of BSPA for motion estimation in video coding. Partial distortion elimination scheme used in EMSEA was improved and then the improved PDE scheme was applied to BSPA. The FBSPA can find the global optimum solution in the same way that FSA can. And FBSPA can reduce the computations of block matching calculation of BSPA 2.9% maximally. FBSPA is a very efficient solution for video

coding applications that require both very low bit-rates and good coding quality.

References

[1] H. G. Musmann, P. Pirsh, and H. J. Gilbert, "Advances in picture coding," Proc. IEEE, Vol. 73, pp. 523-548, Apr. 1985.

[2] J. R. Jain, and A. K. Jain, "Displacement measurement and its application in interframe image coding," IEEE Trans. Commun., Vol. COMM-29, pp. 1799-1808, Dec. 1981.

[3] A. N. Netravali, and J. D. Robbins, "Motion compensated television coding: Part I," Bell Syst. Tech. J, Vol. 58, pp. 631-670, Mar. 1979.

[4] CCITT Standard H.261, "Video codec for audiovisual services at px64 kbit/s," ITU, 1990.

[5] ITU-T DRAFT Standard H.263, "Video coding for narrow telecommunication channel at (below) 64kbit/s," ITU, Apr. 1995.

[6] ISO-IEC JTC1/SC2/WG11, "Preliminary text for MPEG video coding standard," ISO, Aug. 1990.

[7] T. Koga, K. Iinuma, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," Proc. Nat. Telecommunications Conf., pp. G5.3.1-G.5.3.5, Nov. 1981.

[8] A. Puri, H.-M. Hang, and D. L. Schilling, "An efficient block matching algorithm for motion compensated coding," Proc. Int. Conf. Acoust., Speech, Signal Processing, pp. 25.4.1-25.4.4, 1987.

[9] M. Ghanbari, "The cross-search algorithm for motion estimation," IEEE Trans. Commun., Vol. 38, No. 7, pp. 950-953, July 1990.

[10] M. J. Chen, L. G. Chen and T. D. Chiueh, "One-dimensional full search motion estimation algorithm for video coding," IEEE Trans. Circuits Syst. Video Technol., Vol. 4, pp. 504-509, Oct. 1994.

[11] H. M. Jong, L. G. Chen, and T. D. Chiueh, "Accuracy improvement and cost reduction of 3-step search block matching algorithm for video coding," IEEE Trans. Circuits Syst. Video Technol, Vol. 4, pp. 88-91, Feb. 1994.

[12] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," IEEE Trans. Circuits Syst. Video Technol., Vol. 4, pp. 438-442, Aug. 1994.

[13] J. Y. Tham, S. Ranganath, M. Ranganath, and A. Ali Kassim, "A novel unrestricted center-biased diamond search algorithm for block motion estimation," IEEE Trans. Circuits Syst. Video Technol., Vol. 8, No. 4, pp. 369-377, Aug. 1998.

[14] 정원식, 이법기 etc., "이웃 탐색점에서의 평균 절대치 오차 및 탐색 영역 줄임을 이용한 고속 블록 정합 알고리즘", 한국 통신 학회 논문지 '00-1, Vol.25, No1B.

[15] B. Liu, and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," IEEE Trans. Circuits Syst. Video Technol., Vol. 3, No. 2, pp. 148-157, Apr. 1993.

[16] K. H. K. Chow, and M. L. Liou, "Genetic motion search algorithm for video compression," IEEE Trans. Circuits Syst. Video Technol., Vol. 3, pp. 440-446, Dec. 1993.

[17] W. Li, and E. Salari, "Successive elimination algorithm for motion estimation," IEEE Trans. Image Processing, Vol. 4, No.1, pp. 105-107, Jan. 1995.

[18] C.H. Lee, and L.H. Chen, "A Fast Motion Estimation Algorithm Based on the Block Sum Pyramid Algorithm," IEEE Trans. Image Processing, Vol. 6, No. 11, pp. 1587-1591, Nov. 1997.

[19] X. Q. Gao, C.J. Duanmu, and C.R. Zou, "A multilevel successive elimination algorithm for block matching motion estimation," IEEE Trans. Image Processing, Vol. 9, No .3, pp. 501-504, Mar. 2000.

[20] S.M. Jung, S.C. Shin, H. Baik, and M.S. Park, "Efficient Multilevel Successive Elimination Algorithms for Block Matching Motion Estimation," IEE Vision and Image Signal Processing, Vol. 149, No. 2, pp. 73-84, Apr. 2002.

[21] Y.Q. Zhang and S. Zafar, "Predictive Block Matching Motion Estimation for TV Coding-part II: Interframe Prediction," IEEE Trans. Broadcast., Vol. 37, pp. 102-105, Sept. 1991.

[22] A. Gersho and R. M. Gray, "Vector Quantization and

Signal Compressin," Boston, MA: Kluwer, 1991.

정 수 목(Soo-Mok Jung) 정회원

1984년 2월 : 경북대학교 전자공학과 (공학사)
1986년 2월 : 경북대학교 전자 공학과 (공학석사)
2002년 2월 : 고려대학교 컴퓨터학과 (공학박사)
1986년 1월 ~ 1991년 2월 : LG정보통신 연구소 연구원
1988년 3월 ~ 현재 : 삼육대학교 컴퓨터과학과 부교수
<관심분야> : 멀티미디어, 영상처리